CBOR vs. TLS PL

- TLS presentation language
  - Originally used in TLS, but now also in MLS.
  - Not a formal schema or data model language, just specifies structs
  - Extensibility accomplished using types with select statements
  - Not a lot of tooling for decoding, debugging, but it is fairly simple to roll-your-own. No standard debugging format.
  - Already used for MLS and currently for MIMI protocol
  - Lengths of large vectors require bit shifting (not super straightforward in JavaScript for example)
  - Vector lengths are in octets, not units

- CBOR
  - General purpose format for carrying binary data. Historical motivation was for compactness
  - Has a formal data definition language (CDDL) with capabilities comparable to JSON Schema.
  - Has some tooling (heavily weighted toward Ruby), to verify syntax of CDDL, verify basic structure of CBOR, and validate instance documents.
  - Can be converted into JSON (lossy) and several debug formats which are human readable:
    - CBOR pretty printed hex is about the same to read as annotated TLS PL
    - Extended Diagnostic Notation is way easier to read (and generate examples)
    - “Deterministic CBOR” has exactly one representation of any instance document
  - Array/map/string lengths are in units. No bit shifting, but small length items are stuffed together with type code.
**Schema Format 1/2**

- **TLS PL example**

  ```
  struct {
    optional<MessageId> replaces;
    opaque topicId<V>;
    uint32 expires;
    optional<ReplyToInfo> inReplyTo;
    MessageId lastSeen<V>;
    Extension extensions<V>;
    NestablePart body;
} MimiContent;
  
  enum {
    none(0),
    sha256(1),
    (255)
  } HashAlgorithm;

  struct {
    MessageId message;
    HashAlgorithm hashAlg;
    opaque replyToHash<V>;  /* hash of content format */
    inReplyTo;
  }
  
  struct {
    Utf8 name<1..255>;
    opaque value<0..65535>;
  } Extension;
  ```

- **CDDL example**

  ```
  mimiContent = [
    replaces: null / MessageId,
    topicId: bstr,
    expires: uint .size 4,
    inReplyTo: null / InReplyTo,
    lastSeen: [ * MessageId ],
    extensions: { * name => value },
    nestedPart: NestedPart
  ]

  InReplyTo = [
    message: MessageId,
    hashAlg: uint .size 8,
    hash: bstr
  ]

  MessageId = bstr .size 32  ; MessageId is derived from SHA256 hash
  name = tstr .size (1..255)
  value = bstr .size (0..4095)
  ```
Schema Format 2/2

- **TLS PL example**

```c
enum {
    null(0),
    single(1),
    external(2),
    multi(3),
    (255)
} PartCardinality;

struct {
    Utf8 contentType<V> ; /* An IANA media type {8} */
    opaque content<V> ;
} SinglePart;

enum {
    /* {9} */
    chooseOne(0), /* receiver picks exactly one part to process */
    singleUnit(1), /* receiver processes all parts as single unit */
    processAll(2), /* receiver processes all parts individually */
    (255)
} MultiplePartSemantics;

struct {
    Disposition disposition ; /* {10} */
    Utf8 language<V> ; /* {11} */
    uint16 partIndex ; /* {12} */
    PartCardinality cardinality ;
    select(cardinality) {
        case null: ;
        case single: {
            SinglePart part;
        }
        case external:
            ExternalPart part;
        case multi:
            MultiplePartSemantics partSemantics;
            NestablePart parts<V> ;
    }
} NestablePart;

enum {
    unspecified(0),
    render(1),
    reaction(2),
    profile(3),
    inline(4),
    icon(5),
    attachment(6),
    session(7),
    preview(8),
    (255)
} Disposition;

NestedPart = {
    disposition: baseDispos / $extDispos / unknownDispos,
    language: tstr,
    partIndex: uint .size 2,
    (NullPart // SinglePart // ExternalPart // MultiPart)
}

NullPart = ( cardinality: nullpart )

SinglePart = {
    cardinality: single,
    contentType: tstr,
    content: bstr
}

ExternalPart = {
    cardinality: external,
    contentType: tstr,
    url: url,
    expires: uint .size 4,
    size: uint .size 8,
    encAlg: uint .size 2,
    key: bstr,
    nonce: bstr,
    aad: bstr,
    hashAlg: uint .size 1,
    contentHash: bstr,
    description: tstr
}

MultiPart = {
    cardinality: multi,
    partSemantics: chooseOne / singleUnit / processAll,
    parts: [2* NestedPart]
}

baseDispos = {
    unspecified: 0,
    render: 1,
    reaction: 2,
    profile: 3,
    inline: 4,
    icon: 5,
    attachment: 6,
    session: 7,
    preview: 8
}

nullpart = 0
single = 1
external = 2
multi = 3
chooseOne = 0
singleUnit = 1
processAll = 2
unknownDispos = {
    unspecified: 9..255
}
```
Original Message Example

- TLS PL example
- 99 octets

- CBOR example
- 99 octets

```c
/* MimiContent struct */
0x00  optional replaces (present = 0)
0x00  length of topicId
0x00000000  expires
0x00  optional inReplyTo (present = 0)
0x00  length of lastSeen vector
0x00  length of extensions vector
/* NestablePart struct (body)*/
0x01  disposition = render
0x00  length of language
0x0000  partIndex = 0 (1st part)
0x01  cardinality = single part
/* SinglePart struct (part) */
0x1b  length of contentType
 0x746578742f6d61726b646f776e3b6368 /* text/markdown;charset
 61727365743d7974662d38 /* rset=utf-8
0x38  length of content
 0x469206556769677076542072656d /* Hi everyone, we just shipped release 2.0. ___Good work___!
 66727374727570746f6e65642072656d /* just shipped release 2.0. ___Good work___!
 656173520a10e020e285f5f476f6f64 /* text/markdown;charset=utf-8," ease 2.0. ___Good work___!
20776f726b5f5f21

[null,  
 h'',  
 h0, 
 [null,  
 {},  
 [body {NestedPart} {body (NestedPart) }  
 1,  
   disposition = render  
   "text/markdown;charset=rset=utf-8"  
 0,  
   partIndex = 1st part  
 1,  
   cardinality = single part  
   "Hi everyone, we just shipped release 2.0. ___Good work___!"
 ];
]
```
TLS PL example. 238 octets

```c
/* MimiContent struct */
0x01  // optional replaces (present = 1)
0xe701beee59f9376282f39092e1041b2ac2e3aad1776570ca128de244979c71ed
0x00  // length of topicId
0x00000000 expires
/* inReplyTo */
0x01  // optional inReplyTo (present = 1)
0x3c14744d1791d02548232c23d35efa9 // 0x20 octet message ID
0x68174ba385af066011e43bd7e51501 // Original message
0x01  // hashAlg = sha256
0x20  // hash is 0x20 bytes
0x6b44053cb68e3f0cdd219da8d7104afc
2ae5ffff782154524ecf993de39345a5
0x40  // length of lastSeen vector (2 items)
0x4dcab7711a77ea1dd025a6a1a7fe01ab // Reaction
3bd6d90f82417663cb752dfcc37779a1
0x65b0bddd71edc83554aee21380080f4a3 // Mention
ba77985da34528a515f3ac3e8e4998b8
0x00  // length of extensions vector
/* NestablePart struct (body)*/
0x01  // disposition = render
0x00  // length of language
0x0000 partIndex = 0 (1st part)
0x01  // cardinality = single part
/* SinglePart (part) */
0xb  // contentLength
0x746577842f6d61726664f776e3b6368
6172736573d757466d38
0x21  // length of content
0x5269676874206f6e21285f436f666772
6174756c6174696f6e3735f2027616c6c
"Right on! _Congrats_ "atulations_ "all"!"
```

CBOR example. 247 octets

```c
[...
0; # replaces = Reply message
h'ee701beee59f9376282f39092e1041b2ac2e3aad1776570ca128de244979c71ed',
', # topicId
0, # expires = never
[ # InReplyTo
    # message = Original message
h'd3c14744d1791d02548232c23d35efa97668174ba385af066011e43bd7e51501',
1, # hashAlg = sha256
    # hash
h'6b44053cb68e3f0cdd219da8d7104afc2ae5fffff782154524ecf093de39345a5'
    ],
    # lastSeen (2 items)
        # Reaction message
h'4dcab7711a77ea1dd025a6a1a7fe01ab3bd690f82417663cb752dfcc37779a1',
        # Mention message
h'6b50bddd71edc83554ae21380080f4a3ba77985da34528a515fac3c38e4998b8'
],
{...}
    # extensions
    # body (NestedPart)
1, # disposition = render
    # language
    0, # partIndex = 1st part
    # cardinality = single part
"text/markdown; charset=utf-8", # contentType
"Right on! _Congratulations_ y\'all"
] 5
```
Attachment Example

- TLS PL example. 178 octets

```c
/* MimiContent struct */
0x00  optional replaces (present = 0)
0x00  length of topicId
0x00000000 expires is zero
0x00  optional inReplyTo (present = 0)
0x20  length of lastSeen vector
0x5c95a4dfddab84348bccc265a479299fb // Expiring
da3a2eeca3d490985da5113e5480c7f1
0x00  length of extensions vector
/* NestablePart struct (body) */
0x06  disposition = attachment
0x02  length of language
0x656e  "en"
0x0000  partIndex = 0 (1st part)
0x02  cardinality = external part
/* External Part */
0x6a74770733a2f2f6578616d70c652e  "https://example.com/"
0x27  length of url
0x636f6d2f73746f726167652f62696766  "com/storage/bigfile"
0x696c6526d7034  "ile.mp4"
0x0000000000 expires is zero
0x800000002a3c6ced1 size is 708234961 octets
0x0001  encAlg is 0x8001 = AES-128-GCM
0x10  key is 16 octets
0x21399320958a6f4c745d6e670d95e8d8
0x0c  nonce is 12 octets
0xc6cf2c33f21527dd676f5b
0x08  aad is zero octets
0x01  hashAlg = sha256
0x20  content hash is 32 octets
0x9ab1a7a8cf0890baaae7ee016c7312fccc08ba46498389458ee44f0276e783163
0x0c  description is 0x1c octets
0x3220ef6757232066fd62066b5792973  "2 hours of key signing video"  
6976e676e672076696466f  "signing video"
```

- CBOR example. 193 octets

```c
null,  # replaces
"h'`,  # topicId
0,  # expires = never
null,
[
    # inReplyTo
    # lastSeen (1 item)
    # Expiring message
    h'5c95a4dfddab84348bccc265a479299fb3a2eeca3d490985da5113e5480c7f1' ,
],
{},  # extensions
    # body (NestablePart)
    6,  # disposition = attachment
    "en",  # language = en
    0,  # partIndex = 1st part
    2,  # cardinality = external part
    "video/mp4",
    32("https://example.com/storage/bigfile.mp4"),  # url
    0,  # expires
    708234961,
    1,  # size
    h'12399320958a6f4c745d6e670d95e8d8',  # key
    h'c86cf2c33f21527dd676f5b',  # nonce
    1,  # hashAlg = sha256
    h'9ab1a7a8cf0890baaae7ee016c7312fccc08ba46498389458ee44f0276e783163',  # content hash
    "2 hours of key signing video"  # description
```